In the claims

1. (currently amended) A method comprising:

evicting a first memory line currently stored in a cache and storing a second memory line not currently stored in the cache in place of the ~~second~~ first memory line in the cache;

while evicting the first memory line, temporarily storing the second memory line in a buffer, the buffer also storing eviction data regarding the first memory line and data resulting from conversion of the second memory line into a set of concurrently performable actions;

upon eviction of the first memory line, moving the second memory line from the buffer into the cache.

2. (original) The method of claim 1, further initially comprising receiving a transaction relating to the second memory line not currently in the cache.

3. (original) The method of claim 1, further comprising, after temporarily storing the second memory line in the buffer, providing a response indicating that a transaction relating to the second memory line has been performed.

4. (original) The method of claim 1, wherein evicting the first memory line currently stored in the cache comprises inserting the first memory line in an eviction queue of memory lines to be evicted from the cache.

5. (original) The method of claim 1, wherein temporarily storing the second memory line in the buffer comprises temporarily storing the second memory line in a data transfer buffer (DTB).

6. (currently amended) The method of claim 1, further initially comprising converting a transaction to which the second memory line relates into [[a]] the set of concurrently performable actions using a multiple-stage pipeline.

7. (original) The method of claim 6, wherein evicting the first memory line currently stored in the cache comprises inserting the first memory line in an eviction queue of memory lines to be evicted from the cache after the transaction has been converted into the set of concurrently performable actions.

8. (original) The method of claim 6, wherein temporarily storing the second memory line in the existing buffer comprises temporarily storing the second memory line in a data transfer buffer (DTB).

9. (original) The method of claim 6, further comprising, after temporarily storing the second memory line in the buffer, providing a response indicating that the transaction has been performed.

10. (currently amended) A system comprising:

a plurality of processors;

local random-access memory (RAM) for the plurality of processors; and,

at least one controller to manage transactions relative to the local RAM, each controller able to concurrently process the transactions while memory lines to which the transactions relate are being loaded into one or more caches by temporarily storing the memory lines into one or more buffers, the one or more buffers also storing eviction data regarding other memory lines to be evicted from the one or more caches and data resulting from conversion of the memory lines into sets of concurrently performable actions.

11. (original) The system of claim 10, wherein the local RAM is divided into a first memory bank and a second memory bank, and the at least one controller comprises a first controller for managing transactions relative to the first memory bank, and a second controller for managing transactions relative to the second memory bank.

12. (original) The system of claim 10, further comprising a plurality of nodes, a first node including the plurality of processors, the local RAM, and the at least one controller, each other node also including a plurality of processors, local RAM, and at least one controller, the plurality of nodes forming a non-uniform memory access (NUMA) architecture in which each node is able to remotely access the local RAM of other of the plurality of nodes.

13. (currently amended) The system of claim 10, wherein the one or more ~~existing~~ buffers comprise one or more data transfer buffers (DTBs).

14. (original) The system of claim 10, wherein each controller comprises an eviction queue into which other memory lines to be evicted from at least one of the one or more caches is inserted.

15. (currently amended) The system of claim 10, wherein each controller comprises a multiple-stage pipeline to convert the transactions into the sets of concurrently performable actions.

16. (original) The system of claim 10, wherein each controller comprises an application-specific integrated circuit (ASIC).

17. (original) The system of claim 10, wherein the transactions comprise at least one of memory line-related requests and memory line-related responses.

18. (currently amended) An article of manufacture comprising:

a computer-readable medium; and,

means in the medium for processing a transaction while a first memory line to which the transaction relates is being loaded into a cache by evicting a second memory line currently in the cache and by temporarily storing the first memory line into a buffer, the buffer also storing eviction data regarding the second memory line and data resulting from conversion of the first memory line into a set of concurrently performable actions.

19. (original) The article of claim 18, wherein the means moves the first memory line from the buffer into the cache after eviction of the second memory line from the cache.

20. (currently amended) The article of claim 18, wherein the medium is ~~one of~~ a recordable data storage medium ~~and a modulated carrier signal~~.